StarFive 赛昉科技

Using VisionFive IIC to Read SHTC3 Data

with Python

Application Note

Version: 1.1

Date: 2022/07/29

Doc ID: VisionFive-ANEN-010-1.1

Legal Statements

Important legal notice before reading this documentation.

PROPRIETARY NOTICE

Information in this document is provided "as is," with all faults. Contents may be periodically updated or revised due to the product development. Shanghai StarFive Technology Co., Ltd. (hereinafter "StarFive") reserves the right to make changes without further notice to any products herein.

StarFive expressly disclaims all warranties, representations, and conditions of any kind, whether express or implied, including, but not limited to, the implied warranties or conditions of merchantability, fitness for a particular purpose and non-infringement.

StarFive does not assume any liability rising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages.

Contact Us

Address: Room 502, Building 2, No. 61 Shengxia Rd., China (Shanghai) Pilot Free Trade Zone, Shanghai, 201203, China

Website: http://www.starfivetech.com

Email: sales@starfivetech.com(sales) , support@starfivetech.com(support)

Preface

About this guide and technical support information.

About this document

This application note provides steps to use VisionFive's IIC to read SHTC3 data through an example program with Python.

Revision History

Table 0-1 Revision History

Version	Released	Revision
V1.1	2022-07-29	Added "cd" in the codeblock `cd /usr/local/lib64/python3.9/site-packages` to make it a complete command.

Notes and notices

The following notes and notices might appear in this guide:

- **Tip:**
  Suggests how to apply the information in a topic or step.
- **Note:**
  Explains a special case or expands on an important point.
- **Important:**
  Points out critical information concerning a topic or step.
- **CAUTION:**
  Indicates that an action or step can cause loss of data, security problems, or performance issues.
- **Warning:**
  Indicates that an action or step can result in physical harm or cause damage to hardware.

Contents

List of Tables

List of Figures

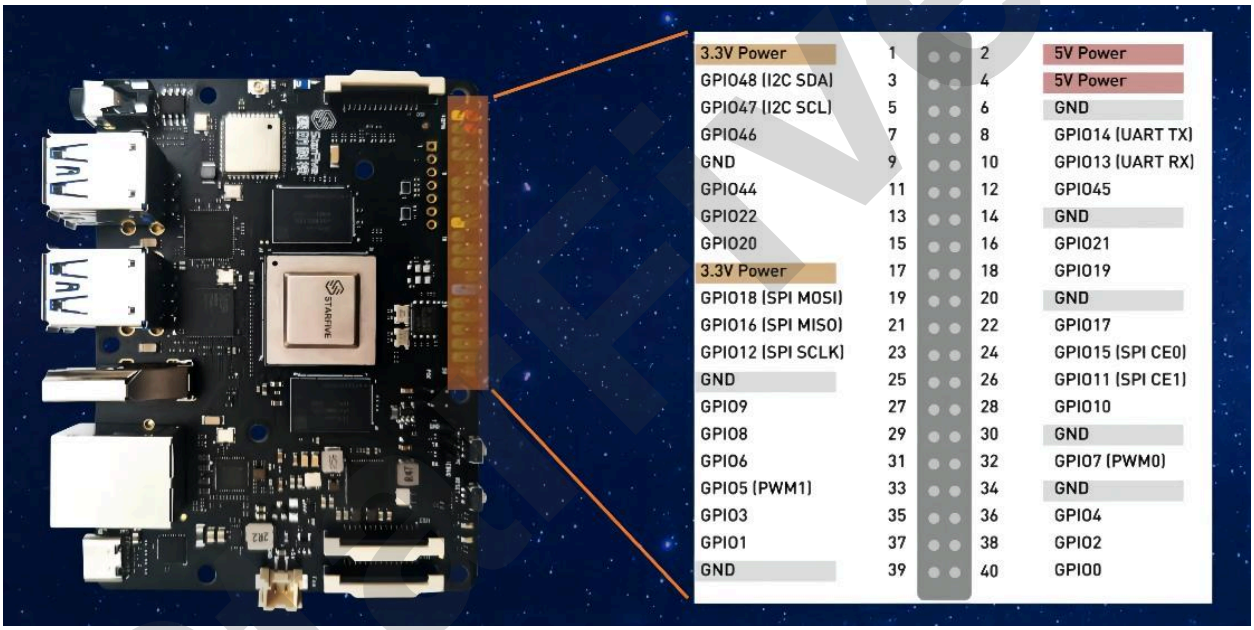
1. Introduction

This application note provides steps to use VisionFive's IIC to read SHTC3 data through an example program with Python.

1.1. 40-Pin Header Definition

The following figure shows the location of the 40-pin header. The VisionFive board is taken as an example:

Figure 1-1 40-Pin Definition



2. Preparation

Before executing the demo program, make sure you prepare the following:

2.1. Preparing Hardware

Prepare the following hardware items before running the demo code:

Table 2-1 Hardware Preparation

Type	M/O*	Item	Notes
General	M	StarFive single board computer	The following boards are applicable: • StarLight • VisionFive
General	M	• 16 GB (or more) micro-SD card • micro-SD card reader • Computer (Windows/MAC/Linux) • USB to serial converter (3.3 V I/O) • Ethernet cable • Power adapter (5 V / 3 A) • USB Type-C Cable	These items are used for flashing Fedora OS into a micro-SD card.
I2C Demo	M	• Sense Hat (B) • Dupont Line	-

Note:
*: M: Mandatory, O: Optional

2.1.1. Hardware Setup

The following table and figure describe how to connect Sense HAT to the 40-pin header:

Table 2-2 Connect Sense Hat (B) to the 40-Pin Header

Sense HAT (B)	40-Pin GPIO Header	
	Pin Number	Pin Name
3V3	1	3.3V Power
GND	9	GND
SDA	3	GPIO48 (I2C SDA)
SCL	5	GPIO47 (I2C SCL)

Figure 2-1 Connect Sense Hat (B) to the 40-Pin Header

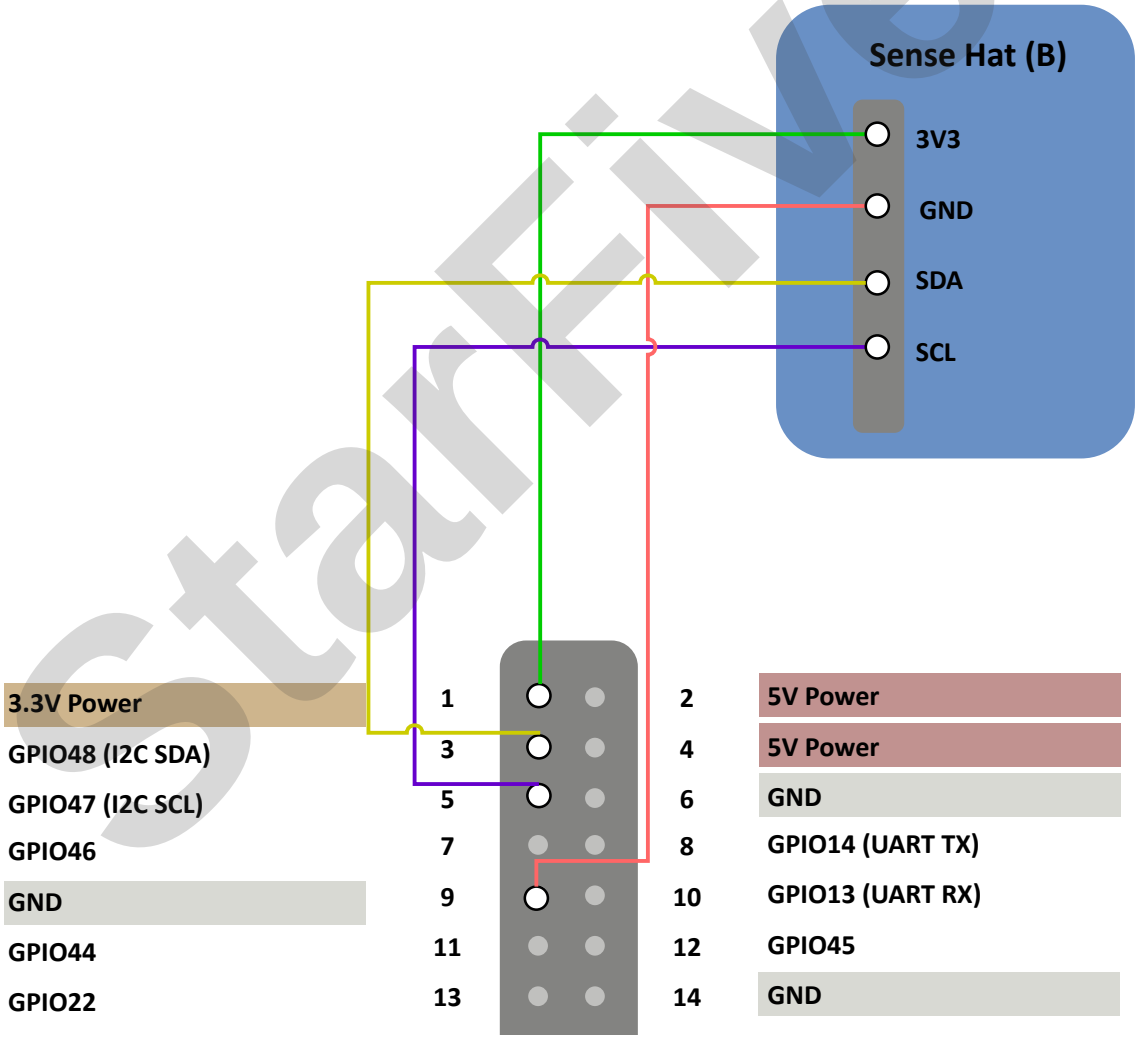
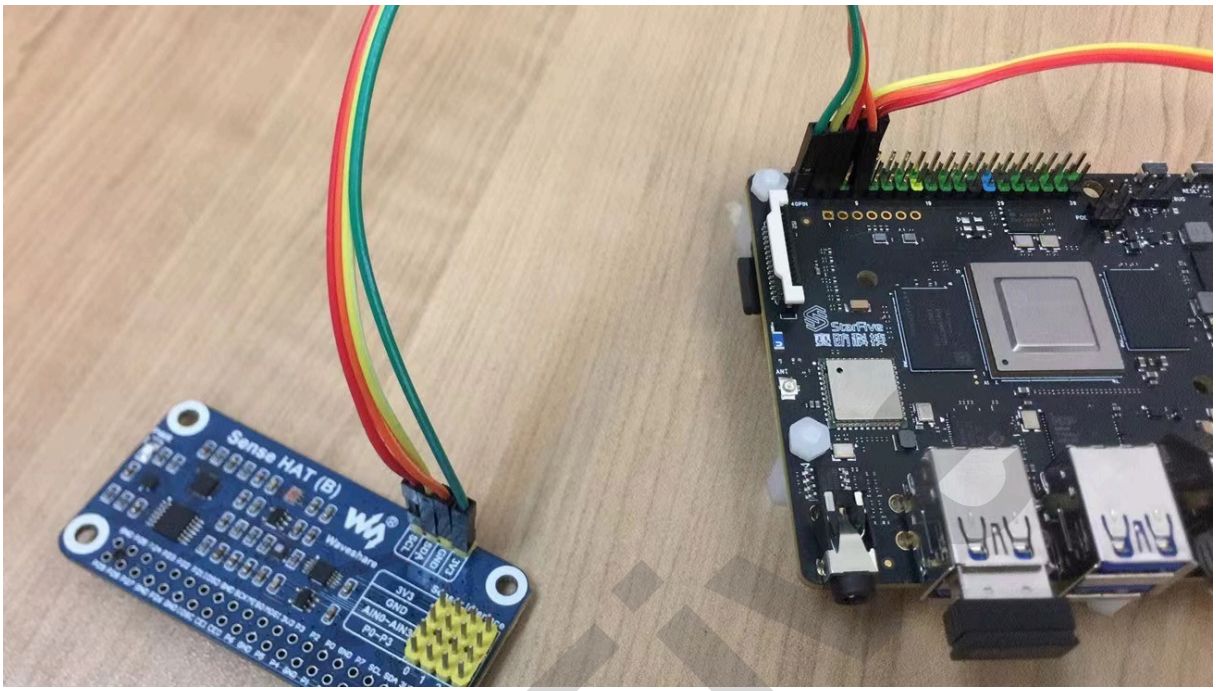


Figure 2-2 Connect Sense Hat (B) to the 40-Pin Header



2.2. Preparing Software

Make sure the following procedures are performed:

1. Flash Fedora OS into a Micro-SD card as described in the *Flashing Fedora OS to a Micro-SD Card* section in the *VisionFive Single Board Computer Quick Start Guide*.
2. Log into the Fedora and make sure VisionFive is connected to the Internet. For detailed instructions, refer to the *Using SSH over Ethernet* or *Using a USB to Serial Converter* section in the *VisionFive Single Board Computer Quick Start Guide*.
3. Execute the `pip` command on VisionFive Fedora to install the `VisionFive.gpio` package:

   ```
   sudo pip install VisionFive.gpio
   ```

   Alternatively, you can execute the following command:

   ```
   sudo pip3 install VisionFive.gpio
   ```

4. (Optional) If you copy the source code to the local directory under VisionFive Fedora, execute the following commands under the source code directory:

   Tip:
   The source code can be downloaded by clicking the following link: VisionFive.gpio.

   ```
   sudo yum install python-devel python3-devel
   sudo python setup.py install
   ```

Alternatively, you can execute the following command:

```
sudo python3 setup.py install
```

3. Running Demo Code

To run the demo code, perform the following on VisionFive Fedora:

1. Locate to the directory where the test code, `I2C_Sense_Hat.py`, exists:

    a. Execute the following command to get the directory where `VisionFive.gpio` exists:

    ```
    pip show VisionFive.gpio
    ```

    Example Result:

    ```
    Location: /usr/local/lib64/python3.9/site-packages
    ```

    Note:

    The actual output depends on how the application is installed.

    b. Execute the following to enter the directory, for example, `/usr/local/lib64/python3.9/site-packages` as indicated in the previous step output:

    ```
    cd /usr/local/lib64/python3.9/site-packages
    ```

    c. Execute the following command to enter the `sample-code` directory:

    ```
    cd ./VisionFive/sample-code/
    ```

2. Under the `sample-code` directory, execute the following command to run the demo code:

    ```
    sudo python I2C_Sense_Hat.py
    ```

    Alternatively, you can execute the following command:

    ```
    sudo python3 I2C_Sense_Hat.py
    ```

    Result:

    The temperature and the humidity data are displayed on the terminal:

    ```
    [riscv@fedora-starfive sample-code]$ sudo python3 I2C_Sense_Hat.py
    i2c_dev: /dev/i2c-1
    Temperature = 27.85°C , Humidity = 56.59 %

    Temperature = 27.83°C , Humidity = 56.60 %

    Temperature = 27.85°C , Humidity = 56.61 %

    Temperature = 27.86°C , Humidity = 56.60 %
    ```

```
Temperature = 27.86°C , Humidity = 56.60 %

Temperature = 27.80°C , Humidity = 56.60 %

Temperature = 27.87°C , Humidity = 56.60 %
```

4. Demo Source Code

The Python source code of this demo is provided for reference purpose only.

`I2C_Sense_Hat.py`:

```
#!/usr/bin/python
'''
Please make sure the sense HAT(B) is connected to the correct pins.
The following table describes how to connect the Sense HAT(B) to the 40-pin
 header.
-----------------------------------------
__Sense HAT (B)___Pin Number_____Pin Name
    3V3             1            3.3 V Power
    GND             9              GND
    SDA             3            I2C SDA
    SCL             5            I2C SCL
-----------------------------------------
'''

import sys
import struct
import fcntl
import os
import math
import time
import VisionFive.i2c as I2C

SHTC3_I2C_ADDRESS = 0x70
I2C_SLAVE = 0x0703
I2C_DEVICE = "/dev/i2c-1"

##Commands
cmd_dict = {
"SHTC3_WakeUp":   0x3517,
"SHTC3_Sleep":    0xB098,
"SHTC3_NM_CE_ReadTH":  0x7CA2,
"SHTC3_NM_CE_ReadRH":  0x5C24,
"SHTC3_NM_CD_ReadTH":  0x7866,
"SHTC3_NM_CD_ReadRH":  0x58E0,
"SHTC3_LM_CE_ReadTH":  0x6458,
"SHTC3_LM_CE_ReadRH":  0x44DE,
"SHTC3_LM_CD_ReadTH":  0x609C,
"SHTC3_LM_CD_ReadRH":  0x401A,
"SHTC3_Software_RES":  0x401A,
"SHTC3_ID":    0xEFC8,
"CRC_POLYNOMIAL":       0x131,
}
```

```
def SHTC3_CheckCrc(data, len, checksum):
    crc = 0xff
    for byteCtr in range(0, len):
        crc ^= data[byteCtr]
        for bit in range(8, 0, -1):
            if(crc & 0x80):
                crc = (crc << 1) ^ cmd_dict["CRC_POLYNOMIAL"]
            else:
                crc = crc << 1
    if (crc != checksum):
        return 1
    else:
        return 0

def SHTC3_WriteCommand(cmd):
    buf0 =  (cmd >> 8)& 0xff
    buf1 = cmd & 0xff
    buf = [buf0, buf1]
    I2C.write(buf)

def SHTC3_WAKEUP():
    SHTC3_WriteCommand(cmd_dict["SHTC3_WakeUp"])
    time.sleep(0.03)

def SHTC3_SLEEP():
    SHTC3_WriteCommand(cmd_dict["SHTC3_Sleep"])

def SHTC_SOFT_RESET():
    SHTC3_WriteCommand(cmd_dict["SHTC3_Software_RES"])
    time.sleep(0.03)

def getdata():
    time.sleep(0.02)
    buf_list = I2C.read(3)
    checksum = buf_list[2]
    DATA = 0
    if (not SHTC3_CheckCrc(buf_list, 2, checksum)):
        DATA = (buf_list[0] << 8 | buf_list[1])
    return DATA

def SHTC3_Read_DATA():
    SHTC3_WriteCommand(cmd_dict["SHTC3_NM_CD_ReadTH"])
    TH_DATA = getdata()
    SHTC3_WriteCommand(cmd_dict["SHTC3_NM_CD_ReadRH"])
    RH_DATA = getdata()
    TH_DATA = 175 * TH_DATA /65536.0 -45.0   #Calculate the temperature
 value.
    RH_DATA = 100 * RH_DATA / 65536.0        #Calculate the humidity value.
```

```
    DATA = [TH_DATA,RH_DATA]
    return DATA

def getTem():
    SHTC3_WriteCommand(cmd_dict["SHTC3_NM_CD_ReadTH"])
    TH_DATA = getdata()
    TH_DATA = 175 * TH_DATA /65536.0 -45.0   #Calculate the temperature
 value.
    return TH_DATA

def getHum():
    SHTC3_WriteCommand(cmd_dict["SHTC3_NM_CD_ReadRH"])
    RH_DATA = getdata()
    RH_DATA = 100 * RH_DATA / 65536.0        #Calculate the humidity value.
    return RH_DATA

def main():
    #Open the Sense HAT by I2C.
    ret = I2C.open(I2C_DEVICE, SHTC3_I2C_ADDRESS)
    if (ret < 0):
        return 0


    SHTC_SOFT_RESET()
    i = 0
    while i < 7:
        Temp = getTem()
        Hum = getHum()
        SHTC3_SLEEP()
        SHTC3_WAKEUP()
        print("Temperature = {:.2f}°C , Humidity = {:.2f} %\n".format(Temp,
 Hum))
        i = i + 1

    I2C.close()
    return 0

if __name__ == "__main__":
    sys.exit(main())
```